Sousanna Chugunova
CMSC 204
Dr.Thai

Write up:

Will your project be able to pass a set of private test cases?

I've extensively tested my project using both the provided GUI tools and specific JUnit tests (*BasicDoubleLinkedList_Test* and *SortedDoubleLinkedList_Test).* These tests cover a wide range of scenarios, including edge cases and typical operations, ensuring that both *BasicDoubleLinkedList* and *SortedDoubleLinkedList* classes perform as expected. Based on this thorough validation and adherence to project specifications, I'm confident that my implementation meets the criteria set by any private test cases that might be used.

Highlight your learning experience and lessons learned

This project taught me a lot about building data structures in Java. I focused on doubly-linked lists and learned how to use iterators for moving through lists and handling exceptions properly. Creating a sorted version of the list also taught me how to use comparators effectively to keep data in order. Overall, I've improved my skills in Java programming and designing organized projects.

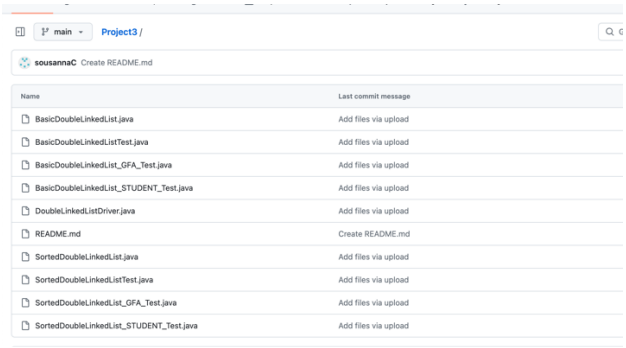
Assumptions that you made

I assumed the provided GUI and tests represented typical situations like I have in the past. For iterators, I followed Java's rules closely, making sure unsupported operations like *remove()* throw the right exceptions. I also relied on the provided comparator to manage sorting in *SortedDoubleLinkedList*, ensuring elements are always in the correct order.

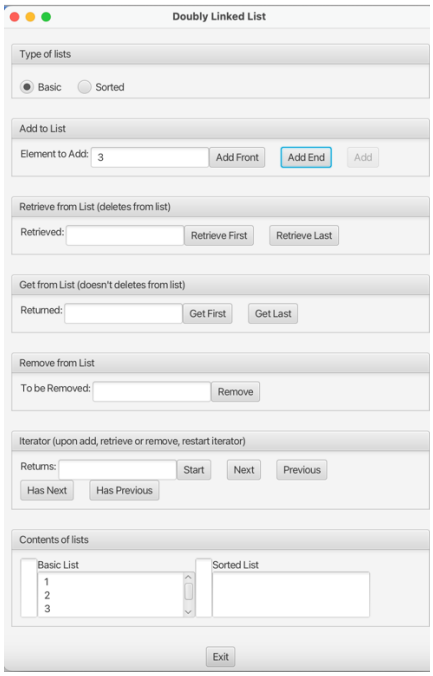
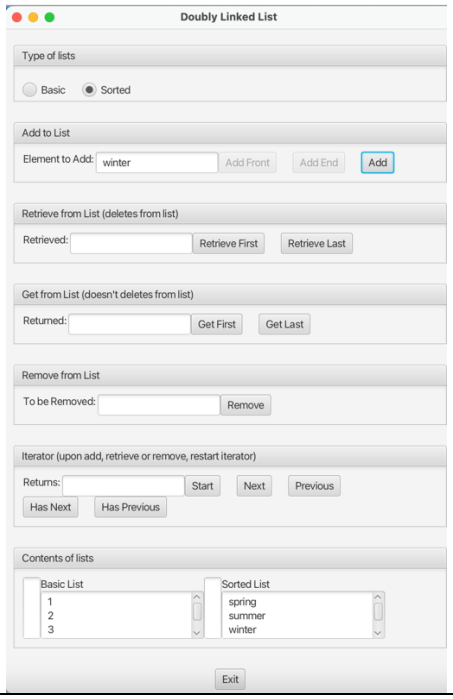
Anything else that I need to know?

All project files, including initial ones from Blackboard, are on my GitHub with clear directories.

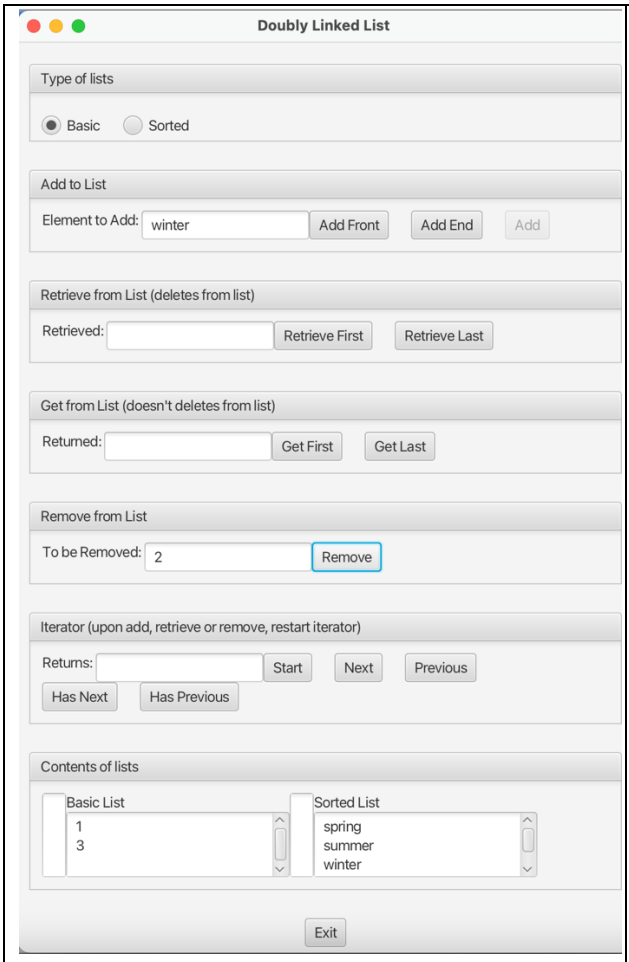
GitHub Screenshot:

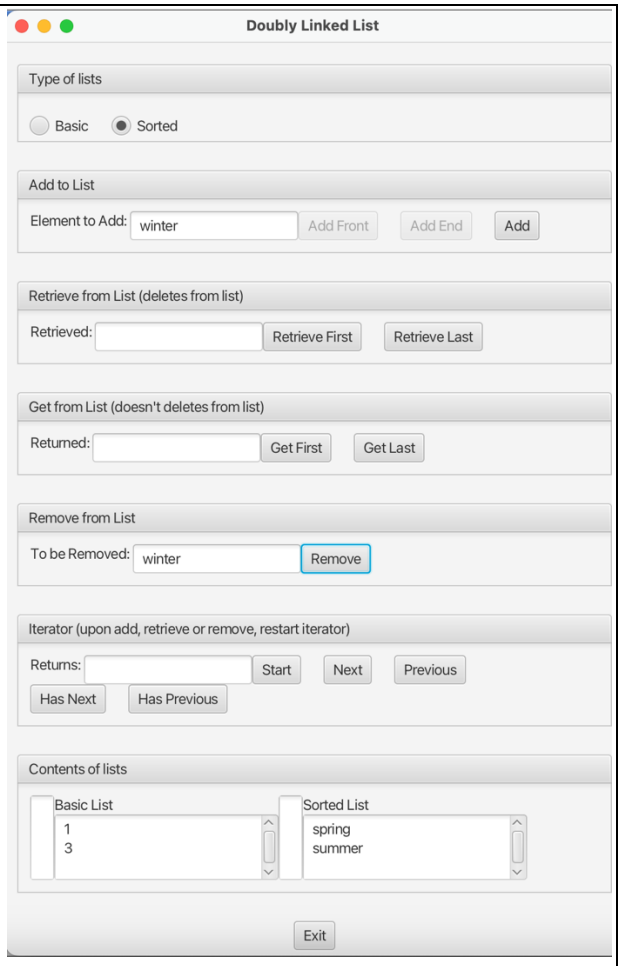


Expected vs Actual Output:

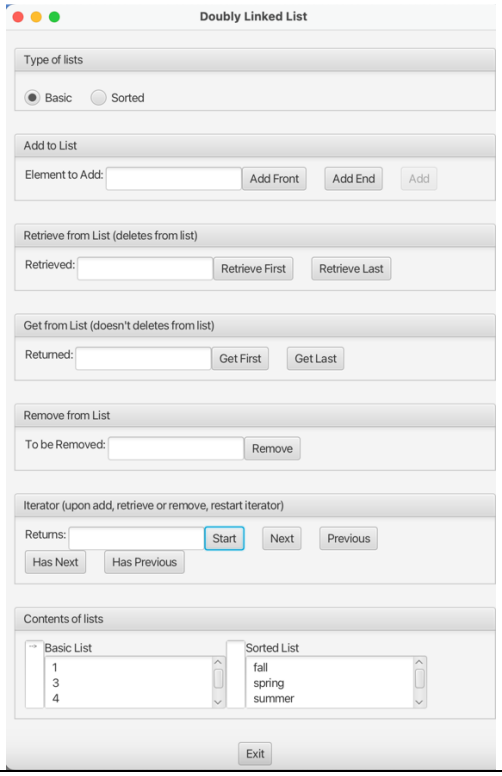
Adding to a basic list	Adding to a sorted list
	

Removing 2 from basic	Removing winter from sorted

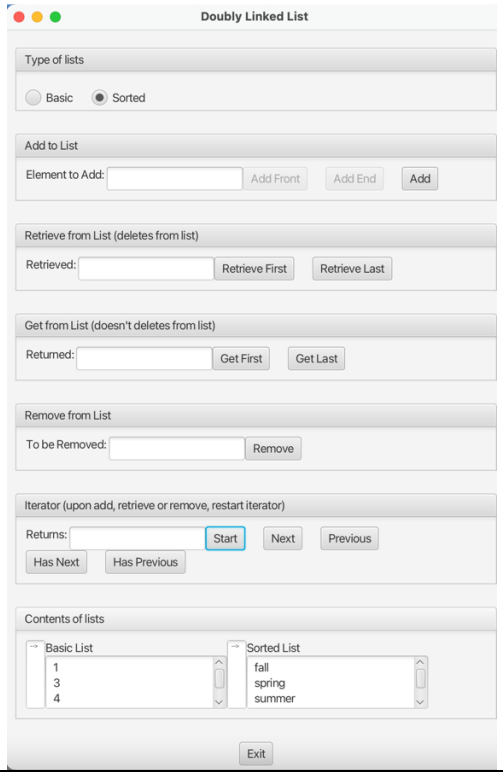

Doubly Linked List
Type of lists
Basic
Sorted
Add to List
Element to Add:
winter
Add Front
Add End
Add
Retrieve from List (deletes from list)
Retrieved:
Retrieve First
Retrieve Last
Get from List (doesn't deletes from list)
Returned:
Get First
Get Last
Remove from List
To be Removed:
2
Remove
Iterator (upon add, retrieve or remove, restart iterator)
Returns:
Start
Next
Previous
Has Next
Has Previous
Contents of lists
Basic List
1
3
Sorted List
spring
summer
winter
Exit


Doubly Linked List
Type of lists
Basic
Sorted
Add to List
Element to Add:
winter
Add Front
Add End
Add
Retrieve from List (deletes from list)
Retrieved:
Retrieve First
Retrieve Last
Get from List (doesn't deletes from list)
Returned:
Get First
Get Last
Remove from List
To be Removed:
winter
Remove
Iterator (upon add, retrieve or remove, restart iterator)
Returns:
Start
Next
Previous
Has Next
Has Previous
Contents of lists
Basic List
1
3
Sorted List
spring
summer
Exit

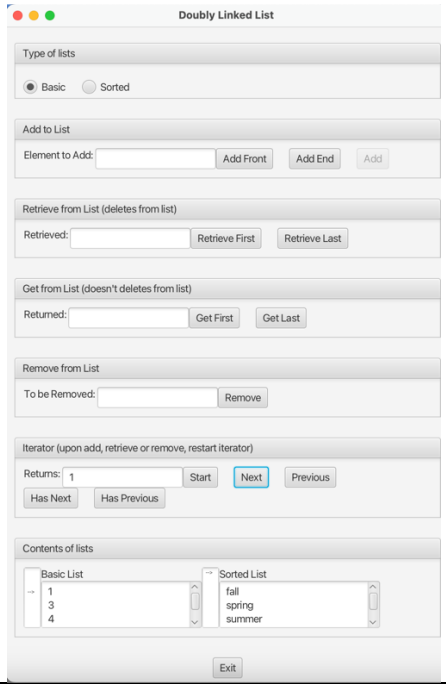
Start the iterators for Basic



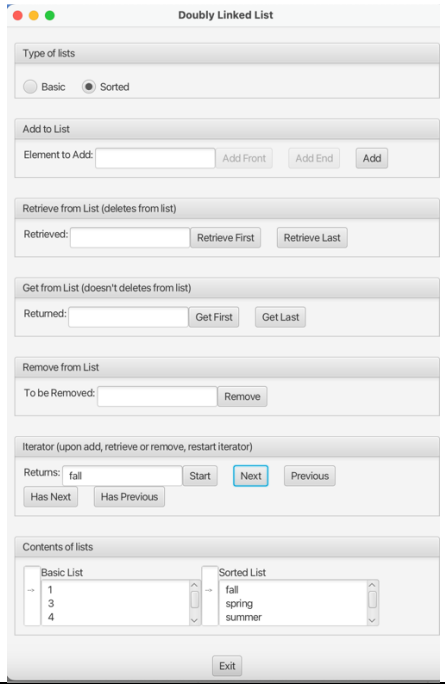
Start the iterators for Sorted



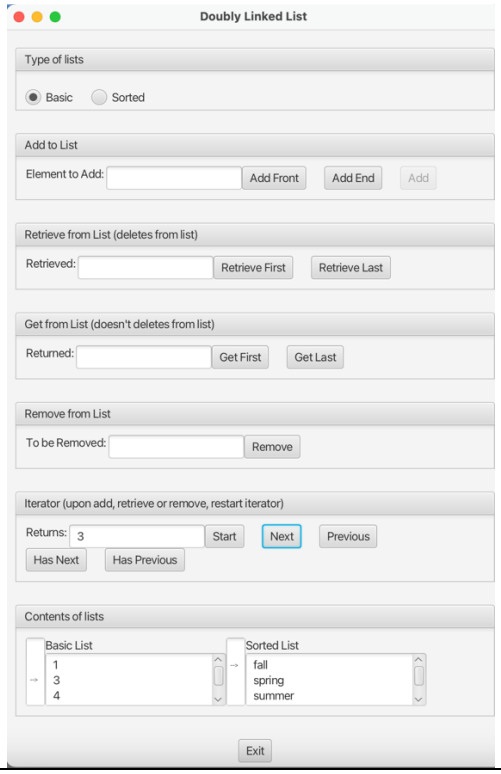
Selecting “Next” for Basic



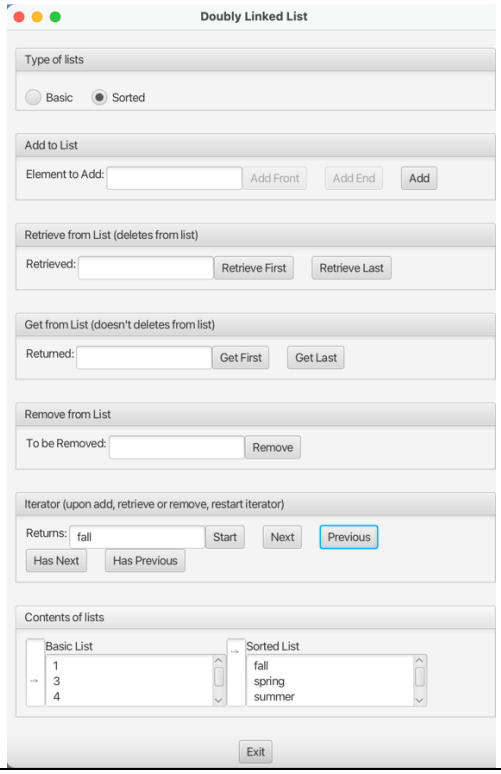
Selecting “Next” for Sorted list



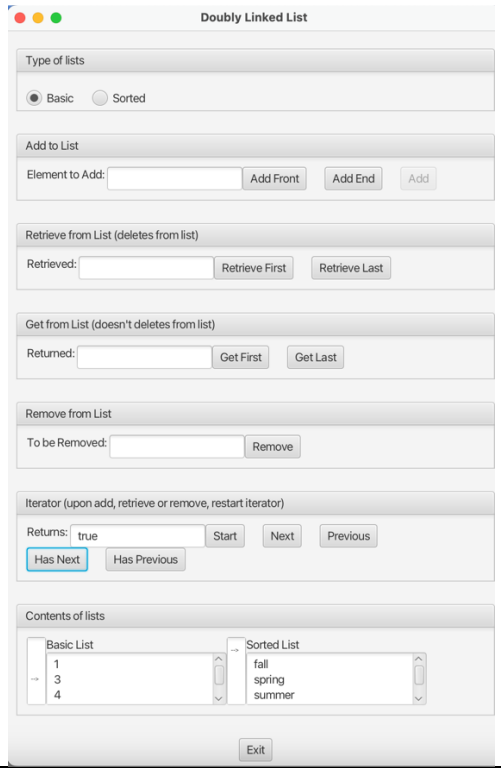
“Next” for basic



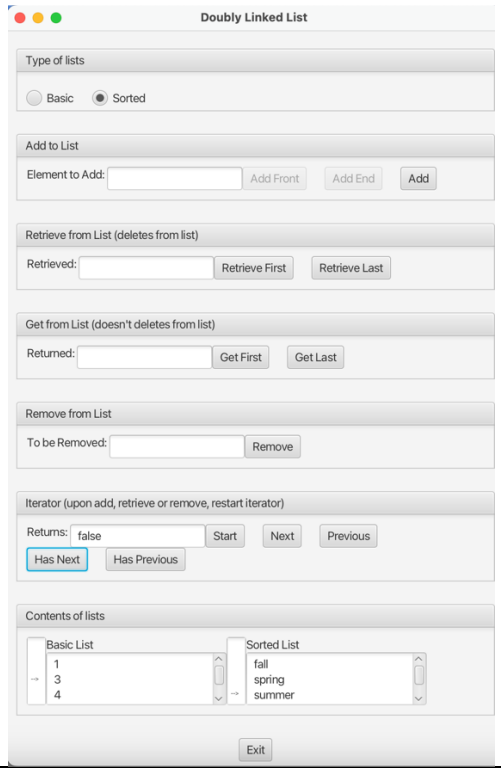
“Previous” for Sorted

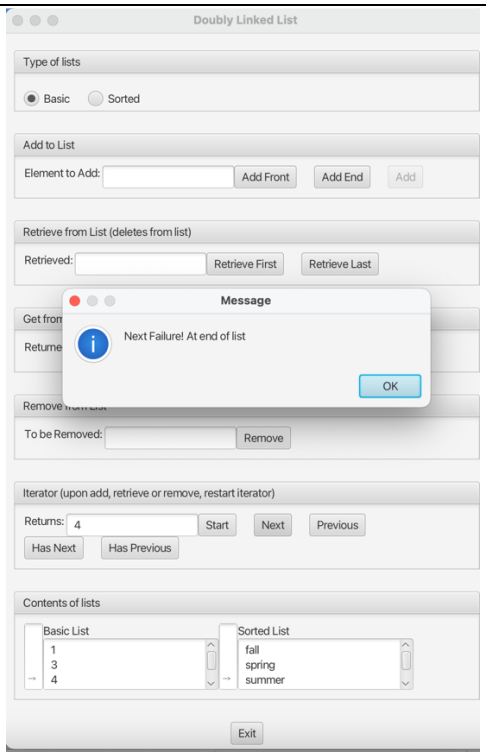


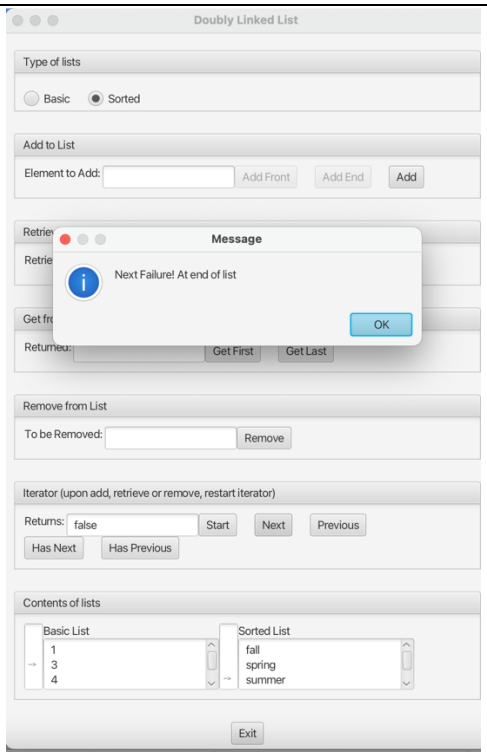
Has next for basic (expected true and error message appears when at the end)



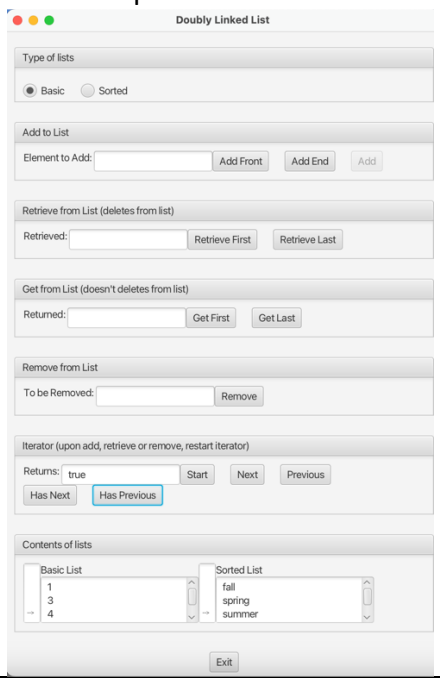
Has Next for sorted (expected false and error message appears)



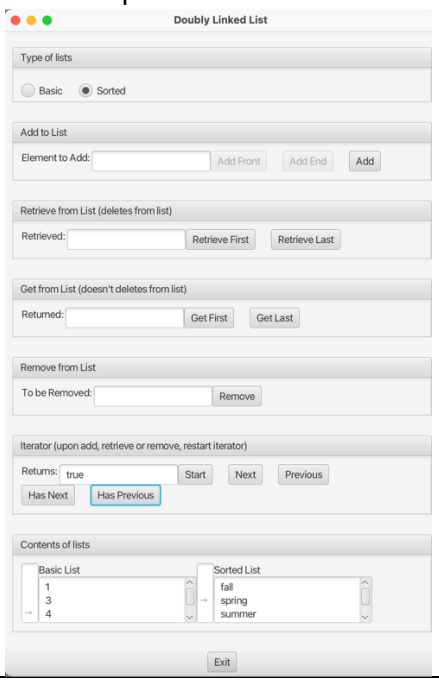




Has previous for basic

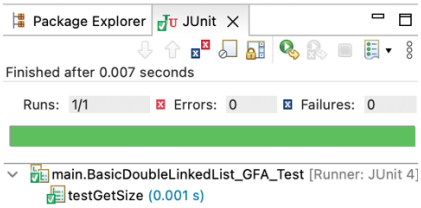


Has previous for sorted

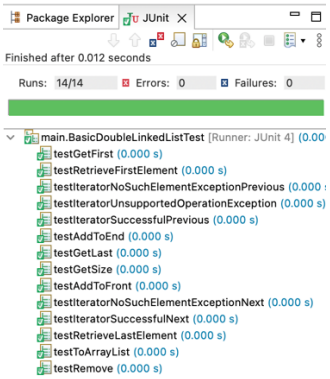


JUNIT Test Cases ran:

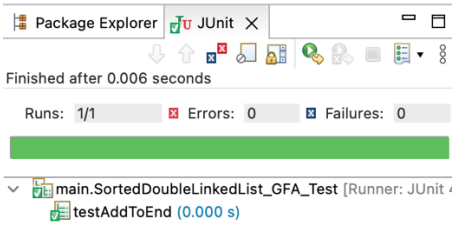
BasicDoubleLinkedList_GFA_Test:



BasicDoubleLinkedListTest:



SortedDoubleLinkedList_GFA_Test:



SortedDoubleLinkedListTest: